PATENT
Attorney Docket No.: SONY-29000

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Application of:	)	Group Art Unit: 2446
	)	
Edward Eytchison *et al.*	)	Examiner: Ali, Farhad
	)	
Serial No.: 10/763,866	)	**APPEAL BRIEF**
	)	
Filed: January 22, 2004	)	162 N. Wolfe Road
	)	Sunnyvale, CA 94086
For: **METHODS AND APPARATUSES FOR AUTOMATIC ADAPTATION OF DIFFERENT PROTOCOLS**	)	(408) 530-9700
	)	Customer No. 28960

Mail Stop Appeal Brief-Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:

In furtherance of the Applicants' Notice of Appeal filed on September 18, 2009, this Appeal Brief is submitted. This Appeal Brief is submitted in support of the Applicants' Notice of Appeal, and further pursuant to the rejection mailed on June 22, 2009, in which Claims 1, 2, 4-11, 13, 15-20 and 22-24 were rejected. The Applicants submit this Appeal Brief to the Board of Patent Appeals and Interferences in compliance with the requirements of 37 C.F.R. § 41.37, as stated in *Rules of Practice Before the Board of Patent Appeals and Interferences (Final Rule)*, 69 Fed. Reg. 49959 (August 12, 2004). The Applicants contend that the rejections of Claims 1, 2, 4-11, 13, 15-20 and 22-24 in this proceeding are in error, were previously overcome and are overcome again by this appeal.

I. REAL PARTIES IN INTEREST

As the assignee of the entire right, title, and interest in the above-captioned patent application, the real parties in interest in this appeal, is:

Sony Corporation, a Japanese corporation
6-7-35 Kitashinagawa, Shinagawa
Tokyo, 141, Japan

Sony Electronics Inc., a corporation of the State of Delaware
1 Sony Drive
Park Ridge, NJ 07656-8003

per the assignment document filed on January 22, 2004.

II. RELATED APPEALS AND INTERFERENCES

The Applicants are not aware of any other appeals or interferences related to the present application.

III. STATUS OF THE CLAIMS

Claims 1, 2, 4-11, 13, 15-20 and 22-24 are pending in this case. Claims 1, 2, 4-11, 13, 15-20, and 22-24 stand rejected under 35 U.S.C. §103(a) as being unpatentable over United States Patent Application Publication No. 2003/0204612 to Warren (hereinafter "Warren", a copy of which is attached as Exhibit A) in view of United States Patent No. 5,623,695 to Lozinski (hereinafter "Lozinski", a copy of which is attached as Exhibit B). Within this Appeal Brief, the rejections of Claims 1, 2, 4-11, 13, 15-20 and 22-24 are appealed.

IV. STATUS OF THE AMENDMENTS FILED AFTER FINAL REJECTION

An Amendment and Response was filed by the appellants on September 18, 2009, in response to the Final Office Action mailed on June 22, 2009. This Amendment and Response contained no amendments to the Specification or the Claims and included only a request for reconsideration in view of the included comments. No Advisory Action or other response was received. Therefore, the claims on appeal are as filed on March 27, 2009 in the Amendment and Response to the Office Action mailed January 6, 2009.

V. SUMMARY OF CLAIMED SUBJECT MATTER

The invention disclosed in the present application number 10/763,866 is directed to methods and apparatuses for translating commands formatted in different protocols into a common application programming interface. The methods and apparatuses detect at least one device; detect a protocol associated with each device; match the protocol with a protocol translator module; and translate a command formatted in the protocol into a translated command formatted in a common application programming interface through the protocol translator module.

The elements of Claim 1, directed to one embodiment of the presently claimed invention, are described in the Specification at least at page 17, line 3 to page 20, line 17, and the accompanying Figures 5-7. The method described there comprises searching for at least one device (110) based on a content type, detecting the at least one device (110), detecting a protocol associated with each device (110), matching the detected protocol with a protocol translator module, and using the protocol translator module to translate a command formatted in the protocol into a translated command formatted in a common application programming interface, wherein the common application programming interface is a single application programming interface that is configured to be used by a plurality of applications (310).

The elements of Claim 7, directed to one embodiment of the presently claimed invention, are described in the Specification at least at page 9, line 3 to page 17, line 2, and the

accompanying Figures 2-4. The system (300) described there comprises means for searching for at least one device (110) based on a content type, means for detecting the at least one device (110), means for detecting a protocol associated with each device (110), means for matching the detected protocol with a protocol translator module, and means for using the protocol translator module to translate a command formatted in the protocol into a translated command formatted in a common application programming interface, wherein the common application programming interface is a single application programming interface that is configured to be used by a plurality of applications (310).

Means for searching for at least one device (110) based on a content type is shown in Figures 3 and 4. Figure 4 is a simplified block diagram illustrating exemplary services, devices and content organized into classes. In one embodiment, these classes are utilized by the system (300) to encapsulate and categorize information corresponding to unique content, devices, or network services that are easily accessed via two or more different network protocols by an illustrative application using a single, common API. These classes include a device manager class (410), a device class (420), and a service class (430). [Present Specification, page 13, lines 7-13] In one embodiment, the device manager class (410) groups devices in response to a GetDeviceByName command that searches the multiple networks for a specific device. [Present Specification, page 14, lines 6-12]

Means for detecting the at least one device (110) is shown in Figure 3. For example, in one instance the software agent (a "control point" application in UPnP terminology) detects UPnP networks by sending out multicast M-search messages to discover devices. Once an available network is detected, the software agent registers (e.g., in the WINDOWS registry) the available network protocol type and the name of the associated translator module, and then the software agent copies the associated translator module from a central repository to an accessible location in preparation for use. [Present Specification, page 4, lines 14-21]

Means for detecting a protocol associated with each device (110) is shown in Figure 4. In one embodiment, these classes are utilized by the system (300) to encapsulate and categorize

information corresponding to unique content, devices, or network services that are easily accessed via two or more different network protocols by an illustrative application using a single, common API. These classes include a device manager class (410), a device class (420), and a service class (430). [Present Specification, page 13, lines 7-13] In one embodiment, the device manager class (430) groups devices in response to a GetDeviceList command that retrieves a list of devices that function using one or more specified network protocols. [Present Specification, page 13, lines 21-23]

Means for matching the detected protocol with a protocol translator module is shown in Figure 3. In other embodiments, the protocol translation layer (350) handles the translation of commands formatted in a plurality of different protocols into translated commands formatted in the common application programming interface. The protocol translation layer (350) supports more than one network protocol. For example, in one instance the protocol translation layer (350) stores more than one translation module for translating commands in multiple different protocols into the common application programming interface. In another instance, the protocol translation layer (350) retrieves an appropriate translation module in response to the protocol to be translated. [Present Specification, page 12, lines 3-11]

Means for using the protocol translator module to translate a command formatted in the protocol into a translated command formatted in a common application programming interface is shown in Figure 3. In some embodiments, the protocol translation layer (350) translates commands utilizing at least one underlying protocol into translated commands utilizing a common application programming interface suitable for use the applications (310), the presentation layer (320), the audio/visual service module (330), and/or the non-audio/visual service module (340). In other embodiments, the protocol translation layer (350) handles the translation of commands formatted in a plurality of different protocols into translated commands formatted in the common application programming interface. [Present Specification, page 11, line 17 to page 12, line 11]

The elements of Claim 8, directed to one embodiment of the presently claimed invention, are described in the Specification at least at page 17, line 3 to page 20, line 17, and the accompanying Figures 5-7. The method of described there comprises searching for at least one service based on a content type, detecting at the least one service, detecting a protocol associated with each service, matching the detected protocol with a protocol translator module, and using the protocol translator module to translate a command formatted in the protocol into a translated command formatted in a common application programming interface, wherein the common application programming interface is a single application programming interface that is configured to be used by a plurality of applications (310).

The elements of Claim 9, directed to one embodiment of the presently claimed invention, are described in the Specification at least at page 17, line 3 to page 20, line 17, and the accompanying Figures 5-7. The method described there comprises searching for a specific device (110) from a plurality of devices (110) based on a content type, detecting the plurality of devices (110) wherein each unique device (110) communicates using a corresponding protocol, displaying an indication of each device (110) if a protocol translator module is matched with the corresponding protocol, and translating a command formatted in the corresponding protocol into a translated command formatted in a common application programming interface through the protocol translator module, wherein the common application programming interface is a single application programming interface that is configured to be used by a plurality of applications (310).

The elements of Claim 17, directed to one embodiment of the presently claimed invention, are described in the Specification at least page 19, line 16 to page 20, line 17, and the accompanying Figure 7. The method described there comprises identifying a plurality of protocol translator modules wherein each protocol translator module is associated with a unique protocol, storing a list representing the plurality of protocol translator modules, displaying an indication of each device (110) having a device protocol that is compatible with one of the plurality of protocol translator modules in the list, and translating a command formatted in the

device protocol into a translated command formatted in a common application programming interface through one of the plurality of protocol translator modules, wherein the common application programming interface is a single application programming interface that is configured to be used by a plurality of applications (310).

The elements of Claim 20, directed to one embodiment of the presently claimed invention, are described in the Specification at least at page 9, line 3 to page 17, line 2, and the accompanying Figures 2–4. The system (300) described there comprises a plurality of applications (310) configured for operating through a single, common application programming interface, a first device (110) configured for operating using a first protocol, a second device (110) configured for operating using a second protocol and a protocol translation layer (350) configured for searching for a first protocol translation module corresponding to the first protocol and for searching for a second protocol translation module corresponding to the second protocol, the first protocol translation module and second protocol translation module stored in a list representing a plurality of protocol translator modules, wherein the protocol translation layer (350) is configured to translate a first command formatted in the first protocol into a command formatted in the single, common application programming interface for use by one of the plurality of applications (310) and to translate a second command formatted in the second protocol into a command formatted in the single, common application programming interface for use by another one of the plurality of applications (310).

The elements of Claim 23, directed to one embodiment of the presently claimed invention, are described in the Specification at least at page 9, line 3 to page 17, line 2, and the accompanying Figures 2-4. The network protocol translation system (300) described there comprises a processor (208, 211) that executes a plurality of run time processes that use only a single application programming interface for network communication, wherein the processor (208, 211) enables at least one of the run time processes to communicate via a first network protocol by executing a first translation module that translates between the first network protocol and the single application programming interface and wherein the processor (208, 211) enables

the at least one of the run time processes to communicate via a second network protocol, different from the first network protocol, and executing a second translation module that translates between the second network protocol and the application programming interface, further wherein the first translation module and second translation module are stored in a list representing a plurality of protocol translator modules.

The elements of Claim 24, directed to one embodiment of the presently claimed invention, are described in the Specification at least at page 17, line 3 to page 20, line 17, and the accompanying Figures 5-7. The method, executed on a computing platform, described there comprises the acts of executing a plurality of run time processes that uses only a single application programming interface for network communication, enabling at least one of the run time processes to communicate via a first network protocol by executing a first translation module that translates between the first network protocol and the single application programming interface and enabling the at least one of the run time processes to communicate via a second network protocol, different from the first network protocol, by executing a second translation module that translates between the second network protocol and the single application programming interface, wherein the first translation module and second translation module are stored in a list representing a plurality of protocol translator modules.

VI. GROUNDS OF REJECTION AND OTHER MATTERS TO BE REVIEWED ON APPEAL

The following issues are presented in this Appeal Brief for review by the Board of Patent Appeals and Interferences:

1. Whether Claims 1, 2, 4-11, 13, 15-20, and 22-24 are properly rejected under 35 U.S.C. §103(a) as being unpatentable over Warren in view of Lozinski.

VII. ARGUMENT

Grounds for Rejection

Within the Office Action, Claims 1, 2, 4-11, 13, 15-20, and 22-24 have been rejected under 35 U.S.C. §103(a) as being unpatentable over Warren in view of Lozinski.

Outline of Arguments

In the discussion that follows, the Applicants discuss the teachings of Warren, the teachings of Lozinski and the teachings of the combination of Warren and Lozinski. As will be discussed in detail below, the combination of Warren and Lozinski does not teach searching for at least one device based on a content type. The combination of Warren and Lozinski also does not teach a single application programming interface that is configured to be used by a plurality of applications. Further, the combination of Warren and Lozinski does not teach storing a list of network protocols available for use, detecting at least one device, detecting a protocol associated with the device, and matching the protocol with a protocol translator.

1. Warren does not teach searching for anything. Further, Warren does not teach searching for at least one device based on content type. Also, Warren does not teach matching the detected protocol with a protocol translator. Moreover, Warren does not teach a single common application programming interface that is configured to be used by a plurality of applications.

Warren teaches an abstraction device with a web services interface. The abstraction device receives web services commands in XML documents, translates the web services commands into one or more device commands in one or more device protocols, and transmits the device commands to one or more network elements. [Warren, ¶¶ 0006, 0024, 0025]. Warren teaches two different implementations of its abstraction device. [Warren, Fig. 1, elements 106a, 106b; Fig. 2, element 206]. In Fig. 1, work manager 102 communicates using a web services protocol, and abstraction device 106 translates between the web services protocol and the protocols used by network elements 108. [Warren, ¶ 0020]. In this embodiment of Warren, the

work manager is an application 102 which utilizes the web services protocol as an applications programming interface. Warren teaches that a network element 108 may be treated as a web service, and that a web service may be published, located, and invoked over a network. [Warren ¶ 0024]. Warren also teaches that the manager [application] 102 may also store information about network elements. [Warren, ¶ 0023].

Warren's second embodiment of the abstraction device shows that web services interface 230 is the top-most layer of the abstraction device 206, further reinforcing that Warren teaches the known web services protocols XML and .NET as its application programming interfaces. [Warren, Fig. 2, elements 206, 230]. Thus, Warren does not teach a *single* application programming interface. In this embodiment, Warren also teaches a plurality of protocol convertors 238a-238e to convert commands 240 to device commands 248, and to convert device responses 268 to alerts 252. [Warren, Fig. 2]. Like the first embodiment, Warren again teaches a plurality of protocol convertors simultaneously loaded into the abstraction device. Warren further teaches that database 236 may store information related to network elements 108. [Warren, ¶¶ 0053-0058]. Such information includes network device IDs 254, network device types 256, command sequence information 258, control instruction information 260, script instruction information 262, field mapping information 264, and registration information. [Warren, ¶¶ 0053-0058]. Warren *does not teach* *searching* for anything. As a result, Warren also does not teach *searching for at least one device based on content type.* Further, as discussed above, Warren teaches that its protocol translators are all always loaded. Warren does not teach *matching the detected protocol with a protocol translator.* Moreover, Warren teaches multiple application programming interface protocols and only a single application: the work manager 102. Also, as recognized in the Office Action of June 22, 2009, Warren does not teach a *single common application programming interface that is configured to be used by a plurality of applications.* [Office Action of June 22, 2009, page 3]

Further, within the Office Action of June 22, 2009, Warren paragraph 0024 is cited as teaching "searching for at least one device based on a content type." [Office Action of June 22,

2009, page 2] The Applicants respectfully disagree. Again, Warren does *not teach searching* for anything. As discussed above, in Warren, the location and type of web service (content) are *published* and are therefore already known. [See Warren ¶ 0024]. As also discussed above, Warren utilizes the information published by web services to populate its database of devices, device types, and other information. [See Warren, ¶¶ 0053-0058]. Nothing in Warren teaches detecting anything by *searching* for it. To the contrary, Warren is silent about network elements which do not publish web services. Warren does not teach *searching for at least one device based on a content type.*

2. Lozinski does not teach searching for at least one device based on a content type. Lozinski also does not teach a single application programming interface that is configured to be used by a plurality of applications. Further, Lozinski does not teach storing a list of network protocols available for use, detecting at least one device, detecting a protocol associated with the device, and matching the protocol with a protocol translator.

Lozinski teaches an application programming interface in a data processing system with multiple communication adapters for a particular service type, such as ISDN. [Lozinski, col. 1, lines 40-46]. Lozinski further teaches that the complex configuration for the end user is avoided because no knowledge of the presence of one manufacturer's product by another is required. [Lozinski, col. 1, lines 65-57]. Thus, Lozinski teaches "translators" for multiple facilities through a common application programming interface (API), but the API is available only to a *single* application, 160. [Lozinski, Fig. 1, 120]. Accordingly, Lozkinski does not teach a single programming interface that is configured to be used by a *plurality* of applications. Lozinski further teaches that a lookup table is used to determine the entry point into the API corresponding to the desired adapter. [Lozinski, col. 1, lines 53-55]. The entry point is determined from information obtained during installation of the adapter. [Lozinski, col. 3, lines 34-45]. Therefore, Lozinski teaches that all translator modules are loaded into the API, and as a result, only *known* devices are included within the scope of its teachings. In other words, Lozinski

teaches devices which are *known* to a common application interface by virtue of being "installed." [Lozinski, col. 3, lines 34-45] Thus, Lozinski does not teach *searching* for at least one device based on a content type. Further, Lozinski does not teach *storing* a list of network protocols available for use, detecting at least one device, detecting a protocol associated with the device, and matching the protocol with a protocol translator.

Moreover, within the Office Action of June 22, 2009, Lozinski is cited as teaching wherein the common application programming interface is a single application programming interface that is configured to be used by a plurality of applications. [Office Action of June 22, 2009, pages 3-4] The Applicants respectfully disagree. As described above, Lozinski teaches *one* application accessing a plurality of similar devices using a common programming interface. [Lozinski, Fig. 1, 120]. Lozkinski does not teach a single programming interface that is configured to be used by a plurality of applications.

3. The combination of Warren and Lozinski does not teach searching for at least one device based on a content type. The combination of Warren and Lozinski also does not teach a single application programming interface that is configured to be used by a plurality of applications. Further, the combination of Warren and Lozinski does not teach storing a list of network protocols available for use, detecting at least one device, detecting a protocol associated with the device, and matching the protocol with a protocol translator.

As discussed above, both Warren and Lozinski teach loading all of their protocol translators. Warren, Lozinski, and their combination, do not teach storing a list of network protocols available for use, detecting at least one device, detecting a protocol associated with the device, and matching the protocol with a protocol translator. As also discussed above, Warren teaches network elements which are known to the abstraction device by virtue of publishing their web services. Similarly, Lozinski teaches devices which are *known* to a common application interface by virtue of being "installed." Thus, Warren, Lozinski, and their combination do not teach *searching for at least one device based on a content type.* As also discussed above,

Warren, Lozinski, and their combination, do not teach *a single application programming interface that is configured to be used by a plurality of applications*.

In contrast to the teachings of Warren, Lozinski and their combination, the presently claimed invention teaches a lightweight, common application programming interface to be used by multiple applications in searching devices containing a type of content. [Present Specification, page 2, lines 14-15]. An application includes a list of available network protocols. In the presently claimed invention, at least one device is *searched for* based on a content type, the at least one device is detected, the protocol associated with the detected device is detected, the detected protocol associated with the device is matched with the device, a protocol translator module is matched with the protocol, *and the protocol translator is loaded*, then the translator module is used to translate a command formatted in the device protocol into a translated command formatted in a common application programming interface that is configured to be *used by a plurality of applications*. [Present Specification, page 11, line 17 to page 17, line 2] Thus, the presently claimed invention creates a lightweight run-time binding by *only loading the protocol translator module(s) which correspond to the at least one device found by virtue of being searched for based on its content.* [Present Specification, page 4, line 2].

4. The claims distinguish over Warren, Lozinski and their combination

The claims are grouped separately below to indicate that they do not stand or fall together.

a. Claim 1

The independent Claim 1 is directed to a method comprising searching for at least one device based on a content type, detecting the at least one device, detecting a protocol associated with each device, matching the detected protocol with a protocol translator module, and using the

protocol translator module to translate a command formatted in the protocol into a translated command formatted in a common application programming interface, wherein the common application programming interface is a single application programming interface that is configured to be used by a plurality of applications. As discussed above, Warren, Lozinski, and their combination do not teach *searching for at least one device based on a content type*. As also discussed above, Warren, Lozinski, and their combination, do not teach detecting at least one device, detecting a protocol associated with the device, and matching the protocol with a protocol translator. Further, as discussed above, Warren, Lozinski, and their combination, do not teach *a single application programming interface that is configured to be used by a plurality of applications*. For at least these reasons, the independent Claim 1 is allowable over the teachings of Warren, Lozinski and their combination.

b. Claims 2 and 4-6

Claims 2 and 4-6 are all dependent on the independent Claim 1. As described above, the independent Claim 1 is allowable over the teachings of Warren, Lozinski and their combination. Accordingly, Claims 2 and 4-6 are all also allowable as being dependent on an allowable base claim.

c. Claim 7

The independent Claim 7 is directed to a system comprising means for searching for at least one device based on a content type, means for detecting the at least one device, means for detecting a protocol associated with each device, means for matching the detected protocol with a protocol translator module, and means for using the protocol translator module to translate a command formatted in the protocol into a translated command formatted in a common application programming interface, wherein the common application programming interface is a

single application programming interface that is configured to be used by a plurality of applications. As discussed above, Warren, Lozinski, and their combination do not teach *searching for at least one device based on a content type*. As also discussed above, Warren, Lozinski, and their combination, do not teach detecting a protocol associated with the device, and matching the protocol with a protocol translator. Further, as discussed above, Warren, Lozinski, and their combination, do not teach *a single application programming interface that is configured to be used by a plurality of applications*. For at least these reasons, the independent Claim 7 is allowable over the teachings of Warren, Lozinski and their combination.

d. Claim 8

The independent Claim 8 is directed to a method comprising searching for at least one service based on a content type, detecting the at least one service, detecting a protocol associated with each service, matching the detected protocol with a protocol translator module, and using the protocol translator module to translate a command formatted in the protocol into a translated command formatted in a common application programming interface, wherein the common application programming interface is a single application programming interface that is configured to be used by a plurality of applications. As discussed above, Warren, Lozinski, and their combination do not teach *searching for at least one device based on a content type*. As also discussed above, Warren, Lozinski, and their combination, do not teach detecting at least one device, detecting a protocol associated with the device and matching the protocol with a protocol translator. Further, as discussed above, Warren, Lozinski, and their combination, do not teach *a single application programming interface that is configured to be used by a plurality of applications*. For at least these reasons, the independent Claim 8 is allowable over the teachings of Warren, Lozinski and their combination.

e. Claim 9

The independent Claim 9 is directed to a method comprising searching for a specific device from a plurality of devices based on a content type, detecting the plurality of devices wherein each unique device communicates using a corresponding protocol, displaying an indication of each device if a protocol translator module is matched with the corresponding protocol, and translating a command formatted in the corresponding protocol into a translated command formatted in a common application programming interface through the protocol translator module, wherein the common application programming interface is a single application programming interface that is configured to be used by a plurality of applications. As discussed above, Warren, Lozinski, and their combination do not teach *searching for at least one device based on a content type*. As also discussed above, Warren, Lozinski, and their combination, do not teach detecting at least one device, detecting a protocol associated with the device and matching the protocol with a protocol translator. Further, as discussed above, Warren, Lozinski, and their combination, do not teach *a single application programming interface that is configured to be used by a plurality of applications*. For at least these reasons, the independent Claim 9 is allowable over the teachings of Warren, Lozinski and their combination.

f. Claims 10, 11, 13, 15 and 16

Claims 10, 11, 13, 15 and 16 are all dependent on the independent Claim 9. As described above, the independent Claim 9 is allowable over the teachings of Warren, Lozinski and their combination. Accordingly, Claims 10, 11, 13, 15 and 16 are all also allowable as being dependent on an allowable base claim.

g. Claim 17

The independent Claim 17 is directed to a method comprising identifying a plurality of protocol translator modules wherein each protocol translator module is associated with a unique protocol, storing a list representing the plurality of protocol translator modules, displaying an indication of each device having a device protocol that is compatible with one of the plurality of protocol translator modules in the list, and translating a command formatted in the device protocol into a translated command formatted in a common application programming interface through one of the plurality of protocol translator modules, wherein the common application programming interface is a single application programming interface that is configured to be used by a plurality of applications. As discussed above, Warren, Lozinski, and their combination, do not teach storing a list of network protocols available for use and displaying an indication of each device having a device protocol that is compatible with one of the plurality of protocol translator modules. Further, as discussed above, Warren, Lozinski, and their combination, do not teach *a single application programming interface that is configured to be used by a plurality of applications*. For at least these reasons, the independent Claim 17 is allowable over the teachings of Warren, Lozinski and their combination.

h. Claims 18 and 19

Claims 18 and 19 are both dependent on the independent Claim 17. As described above, the independent Claim 17 is allowable over the teachings of Warren, Lozinski and their combination. Accordingly, Claims 18 and 19 are both also allowable as being dependent on an allowable base claim.

i. Claim 20

The independent Claim 20 is directed to a system comprising a plurality of applications configured for operating through a single, common application programming interface, a first device configured for operating using a first protocol, a second device configured for operating using a second protocol and a protocol translation layer configured for searching for a first protocol translation module corresponding to the first protocol and for searching for a second protocol translation module corresponding to the second protocol, the first protocol translation module and second protocol translation module stored in a list representing a plurality of protocol translator modules, wherein the protocol translation layer is configured to translate a first command formatted in the first protocol into a command formatted in the single, common application programming interface for use by one of the plurality of applications and to translate a second command formatted in the second protocol into a command formatted in the single, common application programming interface for use by another one of the plurality of applications. As discussed above, Warren, Lozinski, and their combination, do not teach storing a list of network protocols available for use and displaying an indication of each device having a device protocol that is compatible with one of the plurality of protocol translator modules. Further, as discussed above, Warren, Lozinski, and their combination, do not teach *a single application programming interface that is configured to be used by a plurality of applications*. For at least these reasons, the independent Claim 20 is allowable over the teachings of Warren, Lozinski and their combination.

j. Claim 22

Claim 22 is dependent on the independent Claim 20. As described above, the independent Claim 20 is allowable over the teachings of Warren, Lozinski and their combination. Accordingly, Claim 22 is also allowable as being dependent on an allowable base claim.

k. Claim 23

The independent Claim 23 is directed to a network protocol translation system comprising a processor that executes a plurality of run time processes that use only a single application programming interface for network communication, wherein the processor enables at least one of the run time processes to communicate via a first network protocol by executing a first translation module that translates between the first network protocol and the single application programming interface and wherein the processor enables the at least one of the run time processes to communicate via a second network protocol, different from the first network protocol, by executing a second translation module that translates between the second network protocol and the application programming interface, further wherein the first translation module and second translation module are stored in a list representing a plurality of protocol translator modules. As discussed above, Warren, Lozinski, and their combination, do not teach storing a list of network protocols available for use and displaying an indication of each device having a device protocol that is compatible with one of the plurality of protocol translator modules. Further, as discussed above, Warren, Lozinski, and their combination, do not teach *a single application programming interface that is configured to be used by a plurality of applications*. For at least these reasons, the independent Claim 23 is allowable over the teachings of Warren, Lozinski and their combination.

l. Claim 24

The independent claim 24 is directed to a method, executed on a computing platform, comprising the acts of executing a plurality of run time processes that uses only a single application programming interface for network communication, enabling at least one of the run time processes to communicate via a first network protocol by executing a first translation module that translates between the first network protocol and the single application programming

interface and enabling the at least one of the run time processes to communicate via a second network protocol, different from the first network protocol, by executing a second translation module that translates between the second network protocol and the single application programming interface, wherein the first translation module and second translation module are stored in a list representing a plurality of protocol translator modules. As discussed above, Warren, Lozinski, and their combination do not teach executing a plurality of run time processes that uses only a single application programming interface for network communication. For at least these reasons, the independent Claim 24 is allowable over the teachings of Warren, Lozinski and their combination.

5. CONCLUSION

For the above reasons, it is respectfully submitted that the Claims 1, 2, 4-11, 13, 15-20, and 22-24 are allowable over the cited prior art references. Therefore, a favorable indication is respectfully requested.

Respectfully submitted,
HAVERSTOCK & OWENS LLP

Dated: October 30, 2009

By: /Jonathan O. Owens/
Jonathan O. Owens
Reg. No.: 37,902
Attorneys for Applicants

VIII. CLAIMS APPENDIX

This appendix includes a list of the claims under appeal.

1. A method comprising:

searching for at least one device based on a content type;

detecting the at least one device;

detecting a protocol associated with each device;

matching the detected protocol with a protocol translator module; and

using the protocol translator module to translate a command formatted in the protocol into a translated command formatted in a common application programming interface, wherein the common application programming interface is a single application programming interface that is configured to be used by a plurality of applications.

2. The method according to claim 1, further comprising searching for the device from a plurality of devices based on a device identifier.

3. (canceled).

4. The method according to claim 1, further comprising searching for the device from a plurality of devices based on a device type.

5. The method according to claim 1, further comprising searching for the device from a plurality of devices based on a device's availability.

6. The method according to claim 1, further comprising searching for the protocol translator module.

7. A system comprising:

means for searching for at least one device based on a content type;

means for detecting the at least one device;

means for detecting a protocol associated with each device;

means for matching the detected protocol with a protocol translator module; and

means for using the protocol translator module to translate a command formatted in the protocol into a translated command formatted in a common application programming interface, wherein the common application programming interface is a single application programming interface that is configured to be used by a plurality of applications.

8. A method comprising:

searching for at least one service based on a content type;

detecting the at least one service;

detecting a protocol associated with each service;

matching the detected protocol with a protocol translator module; and

using the protocol translator module to translate a command formatted in the protocol into a translated command formatted in a common application programming interface, wherein the common application programming interface is a single application programming interface that is configured to be used by a plurality of applications.

9. A method comprising:

searching for a specific device from a plurality of devices based on a content type;

detecting the plurality of devices wherein each unique device communicates using a corresponding protocol;

displaying an indication of each device if a protocol translator module is matched with the corresponding protocol; and

translating a command formatted in the corresponding protocol into a translated command formatted in a common application programming interface through the protocol translator module, wherein the common application programming interface is a single application programming interface that is configured to be used by a plurality of applications.

10. The method according to claim 9, further comprising detecting the corresponding protocol from each device.

11. The method according to claim 9, further comprising storing the protocol translator module.

12. (canceled).

13. The method according to claim 9, further comprising searching for a specific device from the plurality of devices based on a device identifier.

14. (canceled).

15. The method according to claim 9, further comprising searching for a specific device from the plurality of devices based on a device type.

16. The method according to claim 9, further comprising searching for a specific device from the plurality of devices based on a device's availability.

17. A method comprising:

identifying a plurality of protocol translator modules wherein each protocol translator module is associated with a unique protocol;

storing a list representing the plurality of protocol translator modules;

displaying an indication of each device having a device protocol that is compatible with one of the plurality of protocol translator modules in the list; and

translating a command formatted in the device protocol into a translated command formatted in a common application programming interface through one of the plurality of protocol translator modules, wherein the common application programming interface is a single application programming interface that is configured to be used by a plurality of applications.

18. The method according to claim 17, further comprising searching for additional protocol translator modules.

19. The method according to claim 18, further comprising updating the index in response to the searching for additional protocol translator modules.

20. A system comprising:

a plurality of applications configured for operating through a single, common application programming interface;

a first device configured for operating using a first protocol;

a second device configured for operating using a second protocol; and

a protocol translation layer configured for searching for a first protocol translation module corresponding to the first protocol and for searching for a second protocol translation module corresponding to the second protocol, the first protocol translation module and second protocol translation module stored in a list representing a plurality of

protocol translator modules, wherein the protocol translation layer is configured to translate a first command formatted in the first protocol into a command formatted in the single, common application programming interface for use by one of the plurality of applications and to translate a second command formatted in the second protocol into a command formatted in the single, common application programming interface for use by another one of the plurality of applications.

21. (canceled).

22. The system according to claim 20, further comprising a presentation layer configured for displaying the first device after locating the first protocol translation module.

23. A network protocol translation system comprising:

a processor that executes a plurality of run time processes that use only a single application programming interface for network communication;

wherein the processor enables at least one of the run time processes to communicate via a first network protocol by executing a first translation module that translates between the first network protocol and the single application programming interface; and

wherein the processor enables the at least one of the run time processes to communicate via a second network protocol, different from the first network protocol, by executing a second translation module that translates between the second network protocol and the application programming interface, further wherein the first translation module and second translation module are stored in a list representing a plurality of protocol translator modules.

24. A method, executed on a computing platform, comprising the acts of:

executing a plurality of run time processes that uses only a single application programming interface for network communication;

enabling at least one of the run time processes to communicate via a first network protocol by executing a first translation module that translates between the first network protocol and the single application programming interface; and

enabling the at least one of the run time processes to communicate via a second network protocol, different from the first network protocol, by executing a second translation module that translates between the second network protocol and the single application programming interface, wherein the first translation module and second translation module are stored in a list representing a plurality of protocol translator modules.

IX. EVIDENCE APPENDIX

STATEMENT

Pursuant to 37 C.F.R. § 41.37(c)(1)(ix), the following is a statement setting forth where in the record the evidence of this appendix was entered by the examiner:

Evidence Description:	**Where Entered:**
U.S. Pat. No. 5,623,695	Office Action mailed January 6, 2009
U.S. Pat. Pub. No. 2003/0204612	Office Action mailed October 5, 2007
Office Action mailed June 22, 2009	Examiner Office Action

X. RELATED PROCEEDINGS APPENDIX

There are no related proceedings.